IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Ex Parte Kevin Marshall

Application for Patent: 10/052,784

Filed: November 2, 2001

Group Art Unit 2192

Examiner KENDALL, Chuck O.

For:

METHODS AND APPARATUS FOR DETERMINING SOFTWARE COMPONENT SIZES ASSOCIATED WITH ERRORS

APPEAL BRIEF

BEYER WEAVER & THOMAS, LLP
P.O. Box 70250
Oakland, CA 94612-0250
Attorneys for Appellant

TABLE OF CONTENTS

1. REAL PARTY IN INTEREST

[37 CFR 41.37(c)(1)(i)]

The real party in interest is Sun Microsystems, Inc.

2. RELATED APPEALS AND INTERFERENCES

[37 CFR 41.37(c)(1)(ii)]

There are no related appeals or interferences.

3. STATUS OF CLAIMS

[37 CFR 41.37(c)(1)(iii)]

The following claims have been rejected and appealed: claims 1, 2, 4-5, 7-14 and 16-24.

The following claims have been cancelled: 3, 6, and 15.

The claims on appeal are reproduced below in the Appendix at Section 9 of this Appeal Brief.

4. STATUS OF AMENDMENTS

[37 CFR 41.37(c)(1)(iv)]

No amendments were filed subsequent to final rejection.

5. SUMMARY OF CLAIMED SUBJECT MATTER

[37 CFR 41.37(c)(1)(v)]

5.1. Independent Claim 1

Claim 1 is directed to a method of automatically generating data regarding errors in a software system. Contents of one or more files are examined, where the one or more files examined comprises a history of one or more errors generated during execution of the software system. For example, the one or more files examined may include a workspace history file, an example of which is shown in Fig. 2B. See, e.g., the specification at page 12, lines 3-9, which state:

> FIG. 2B shows an example of the information contained within a sample workspace history file. The "BEGIN COMMENT" entry (222) is used in this

embodiment as an indication that the comment section that follows may contain error IDs which here are 7 digit numbers (224 and 226). The "END Comment" (228) statement is interpreted by the method as a signal that no more error IDs remain in the section. The method further detects commands such as "update" (230), "create" (231), or "rename"(232) as indications that files that have changed follow.

Fig. 2A, for example, is a flowchart illustrating a method for determining a function associated with an error. See, e.g., the specification at page 11, line 5 to page 12, line 2. There, it is discussed how the comments section of the workspace history 104 is processed to determine errors which have resulted from executing the software system.

Fig. 2C is a flowchart illustrating details as to the method for determining a function associated with an error illustrated in Fig. 2A. See, e.g., the specification at page 12, line 10 to page 13, line 15, where it is discussed how character searches may be employed in the Fig. 2A method.

Figs. 3A, 4 and 5 are flowcharts illustrating methods for determining a function associated with an error. Fig. 3A is a flowchart illustrating the correlation of an error ID with a file revision. See, e.g., page 14, line 3 to page 15, line 10. Fig. 4 illustrates a method of identifying the lines changed in a file revision. See, e.g., page 16, line 1 to page 17, line 11. Fig. 5 is a flowchart illustrating the matching of functions with line changes. See, e.g., page 17, line 12 to page 18, line 6.

Thus, for example, in accordance with the methods illustrated in the flowcharts mentioned above and described in the specification, a determination may be made of software components responsible for errors based on examining contents of files that comprise a history of one or more errors generated during execution of the software.

In addition, a size is determined of the one or more software components responsible for the errors. See, e.g., step 112 in Fig. 1.

5.2. Independent Claim 22

Independent claim 22 is similar to independent claim 1, discussed above, except that claim 22 is directed to a computer-readable medium, not a method. Given the similarities of claim 22 to claim 1, a separate concise explanation specifically directed to claim 22 is not provided. Rather, Applicant incorporates the concise explanation provided above regarding claim 1.

5.3. Independent Claim 23

Independent claim 23 is similar to independent claim 1, discussed above, except that claim 23 is directed to an apparatus, not a method. Given the similarities of claim 23 to claim 1, a separate concise explanation specifically directed to claim 23 is not provided. Rather, Applicant incorporates the concise explanation provided above regarding claim 1.

5.4. Independent Claim 24

Independent claim 24 is similar to independent claim 1, discussed above, except that claim 24 is directed to an apparatus including a processor and a memory, where at least one of the processor and the memory is adapted for performing a method similar to that recited in claim 1. Claim 24 is not directed to the method itself. Given the similarities of claim 24 to claim 1, a separate concise explanation specifically directed to claim 24 is not provided. Rather, Applicant incorporates the concise explanation provided above regarding claim 1.

6. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

[37 CFR 41.37(c)(1)(vi)]

6.1. Ground I

Claims 1-2, 16 and 22-24 are rejected under 35 U.S.C. 103(a) as being unpatentable by US Patent No. 6745348 to Chung, in view of US Patent No. 5761510 to Smith.

6.2. Ground II

Claims 4-5 are rejected under 35 USC 103(a) as being unpatentable over Chung in view of Smith, and further in view of US Patent No. 6665824 to Ruhlen.

6.3. Ground III

Claims 7-9 and 13-14 are rejected under 35 USC 103(a) as being unpatentable over Chung in view of Smith in view of Ruhlen, and further in view of US Patent No. 6769114 to Leung.

6.4. Ground IV

Claims 10-12 and 17-21 are rejected under 35 USC 103(a) as being unpatentable over Chung in view of Smith in view of Ruhlen, and further in view of US Patent No. 5946493 to Hanson.

7. ARGUMENT

[37 CFR 41.37(c)(1)(vii)]

7.1. Ground I

7.1.1. Claim 1

Claim 1 is a method claim and includes the feature of:

> examining contents of one or more files indicating one or more errors in the software system to determine ...

The one or more files are recited as

> wherein the one or more files examined comprises a history of one or more errors generated during execution of the software system.

Significantly, by contrast, the "errors" discussed in the Chung primary reference are errors identified by scanning source code of a program, and Chung has nothing whatsoever to do with examining files that comprise a history of one or more errors "generated during execution of the software program."

Claim 1 clearly refers to "history of one or more errors generated during execution of the software program." (emphasis added) Claim 1 does not refer to scanning a file for prospective errors, as disclosed by Chung. Rather, the file (i.e. the contents of the file) is examined for errors that have already occurred. Furthermore, the "scanning" by Chung is to identify errors solely for the purpose of estimating the total number of errors. The real work in Chung, of identifying and fixing all the errors, begins only where the Chung disclosure leaves off.

Clearly, Chung's motivation in estimating the number of errors is "estimating the number of software developers to be assigned to test and debug the particular internationalized software program." See Abstract.

The Smith secondary reference, on the other hand, discloses actual testing and debugging programs for identifying errors generated during execution. With respect to this broad aspect, Smith discloses nothing more than programmers have been doing since the beginning of programming time.

At best, one of ordinary skill in the art would be motivated to combine Chung and Smith to estimate the number of software developers to be assigned to test and

debug an internationalized software program (Chung) and, then, actually test and debug the software program (Smith). This is not what is recited in claim 1.

The Examiner contends that:

> It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of Smith into that of Chung for the inclusion of identifying "errors generated during execution" of the software program. And the motivation for doing so would have been to ensure that library program functions associated with the program can be called dynamically (i.e., during execution) and accessed by other programs written in other languages and perform as they were programmed to performed [sic].

As alleged support for this contention, the Examiner cites to the following portions of Smith – col. 1, line 60 to col. 2, line 5; col. 2, line 67 to col. 3, line 30; and col. 3, line 59 to line 65.

Applicant agrees that these portions of Smith suggest debugging a software program by identifying errors generated during execution. However, there is nothing in Smith to suggest modifying the Chung disclosure to examine files, "wherein the one or more files examined comprises a history of one or more errors generated during execution of the software system." That is, Chung teaches away from such a modification since, by its terms, Chung is concerned with estimating errors prior to debugging a program for the purpose of estimating manpower to later debug the program. Furthermore, Smith is concerned with identifying and fixing errors while debugging a program.

7.1.2. Claims 2, 3, 16, 22, 23 and 24

These claims incorporate the Ground I rejection of claim 1. It is respectfully submitted that the rejection of these claims is improper for at least the reasons that the rejection of claim 1 is improper.

7.2. Ground II

7.2.1. Claims 4-5

Ground II incorporates the Ground I rejection, and further applies Ruhlen for its alleged disclosure of tracking/counting errors which occur during execution of the software components.

At best, Ruhlen would be combined with Chung and Smith as a particular method of debugging (Smith), after estimating the amount of manpower required to fix the errors (Chung).

7.3. Ground III

Ground III incorporates the Ground II rejection, and further applies Leung. It is respectfully submitted that the Ground III rejection is improper for at least the reason that the Ground II rejection is improper.

7.4. Ground IV

Ground IV also incorporates the Ground II rejection. It is respectfully submitted that the Ground IV rejection is improper for at least the reason that the Ground II rejection is improper.

8. CONCLUSION

In view of the foregoing, it is respectfully submitted that the Examiner's rejections (Grounds I to IV) are erroneous. Accordingly, the rejection of the claims should be reversed.

Respectfully submitted,

/ASH/
Alan S. Hodes
Registration No. 38,185

BEYER, WEAVER & THOMAS LLP
Attorneys for Appellant

9. CLAIMS APPENDIX

[37 CFR 41.37(c)(1)(viii)]

CLAIMS ON APPEAL

1. (Previously Presented) A method of automatically generating data regarding errors in a software system, the software system including one or more software components, the method comprising:

examining contents of one or more files indicating one or more errors in the software system to determine one or more of the software components responsible for the errors and a number of the errors attributed to each of the software components determined to be responsible for the errors, wherein the one or more of the files examined comprises a history of one or more errors generated during execution of the software system; and

determining a size of the one or more software components responsible for the errors.

2. (Original) The method as recited in claim 1, further comprising correlating the size of the determined software components with the number of errors attributed to the determined software components, thereby enabling data indicating a probability of errors occurring during execution of a set of software components to be generated from the determined size of the software components determined to be responsible for the errors and the number of the errors attributed to each of the software components determined to be responsible for the errors.

3. (Cancelled)

4. (Previously Presented) The method as recited in claim 1, wherein the contents of one or more files examined further indicates one or more source code modifications made in response to the errors.

5. (Original) The method as recited in claim 4, wherein determining from the one or more files one or more of the software components responsible for the errors comprises:

determining from the source code modifications one or more software components modified to correct the errors.

6. (Cancelled)

7. (Original) The method as recited in claim 1, wherein examining contents of one or more files indicating one or more errors in the software system comprises generating a list of one or more errors corresponding to source code changes and a list of one or more files associated with successful attempts to correct the errors.

8. (Original) The method as recited in claim 7, wherein examining contents of one or more files further comprises correlating a file in the list of files associated with successful attempts to correct the errors with one of the source code changes corresponding to at least one of the list of errors.

9. (Original) The method as recited in claim 8, wherein the correlating a file in the list of files associated with successful attempts to correct the errors with

one of the source code changes corresponding to at least one of the list of errors comprises examining an individual file history for at least one file in the list of files.

10. (Previously Presented) The method as recited in claim 1, wherein determining a size of the one or more software components responsible for the errors comprises determining start and end lines of a section of code modified to fix an error.

11. (Original) The method as recited in claim 10, further comprising converting the start and end lines of a section of code modified to the start and end lines of a current version of a file

12. (Previously Presented) The method as recited in claim 10, further comprising determining the one or more software components responsible for the errors from current versions of the one or more files.

13. (Original) The method as recited in claim 7, wherein the list of files contains information identifying the version of the file and one or more identifiers to identify one or more errors associated with the version of the file.

14. (Original) The method as recited in claim 13,wherein the errors corresponding to source code changes are matched against one or more identifiers associated with a file version.

15. (Cancelled)

16. (Original) The method as recited in claim 1, further comprising:

identifying one or more files responsible for the errors;

obtaining one or more file histories associated with the files responsible for the errors; and

ascertaining from the file histories the one or more software components responsible for the errors.

17. (Original) The method as recited in claim 16, wherein ascertaining from the file histories the one or more software components responsible for the errors further comprises:

ascertaining from the file histories one or more line numbers associated with the software components responsible for the errors.

18. (Original) The method as recited in claim 17, wherein ascertaining from the file histories one or more line numbers associated with the software components responsible for the errors comprises:

matching one or more line numbers associated with modified source code against compiled information associated with the source code.

19. (Original) The method as recited in claim 18, further comprising:

determining the one or more line numbers associated with the modified source code from one or more error identifiers present in the file histories.

20. (Original) The method as recited in claim18, further comprising:

comparing information associated with one or more versions of the source code to determine the one or more line numbers associated with the modified source code.

21. (Original) The method as recited in claim 17, wherein the one or more line numbers associated with the software components responsible for the errors includes a start line and an end line associated with the software components responsible for the errors, wherein determining a size of the software components responsible for the errors is performed from the start line and the end line associated with the software components responsible for the errors.

22. (Previously Presented) A computer-readable medium storing thereon instructions for automatically generating data regarding errors in a software system, the software system including one or more software components, comprising:

instructions for examining contents of one or more files indicating one or more errors in the software system to determine one or more of the software components responsible for the errors and a number of the errors attributed to each of the software components determined to be responsible for the errors, wherein the one or more of the files examined comprises a history of one or more errors generated during execution of the software system; and

instructions for determining a size of the one or more software components responsible for the errors.

23. (Previously Presented) An apparatus for automatically generating data regarding errors in a software system, the software system including one or more software components, comprising:

means for examining contents of one or more files indicating one or more errors in the software system to determine one or more of the software components

responsible for the errors and a number of the errors attributed to each of the software components determined to be responsible for the errors, wherein the one or more of the files examined comprises a history of one or more errors generated during execution of the software system; and

means for determining a size of the one or more software components responsible for the errors.

24. (Previously Presented) An apparatus for automatically generating data regarding errors in a software system, the software system including one or more software components, the apparatus comprising:

a processor; and

a memory, at least one of the processor and the memory being adapted for:

examining contents of one or more files indicating one or more errors in the software system to determine one or more of the software components responsible for the errors and a number of the errors attributed to each of the software components determined to be responsible for the errors, wherein one or more of the files examined comprises a history of one or more errors generated during execution of the software system; and

determining a size of the one or more software components responsible for the errors.

10. EVIDENCE APPENDIX

[37 CFR 41.37(c)(1)(ix)]

No evidence has been submitted pursuant to §§ 1.130, 1.131, or 1.132 of 37 CFR, nor has any other evidence been entered by the examiner.

11. RELATED PROCEEDINGS APPENDIX

[37 CFR 41.37(c)(1)(x)]

There have been no decisions rendered by a court or the Board in any proceeding identified pursuant to paragraph (c)(1)(ii) of 37 CFR 41.37(c)(1).